Universidade Federal de Ouro Preto

Atividades Eletrônica

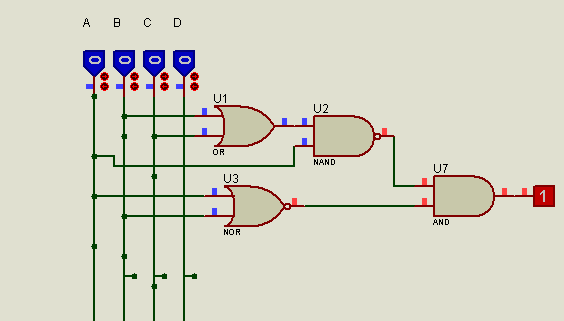
Conteúdo relativo à P2

Fábio Henrique Alves Fernandes

Ouro Preto, 6 de julho de 2019

**Questão 1**

A) 𝑆 = ((~(𝐴 ∙ (𝐵 + 𝐶))) ∙ 𝐷) ∙ (~(𝐴 + 𝐵))

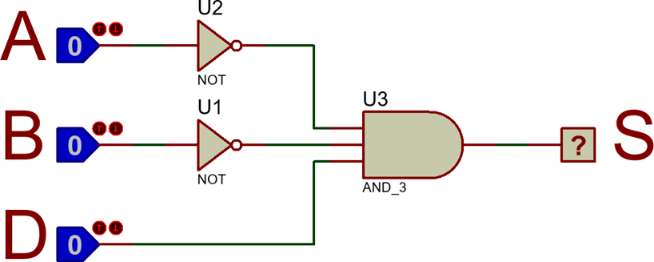


𝑆 = ((~(𝐴 + (~𝐵 ∙ ~𝐶))) ∙ 𝐷) ∙ (~𝐴 ∙ ~𝐵)

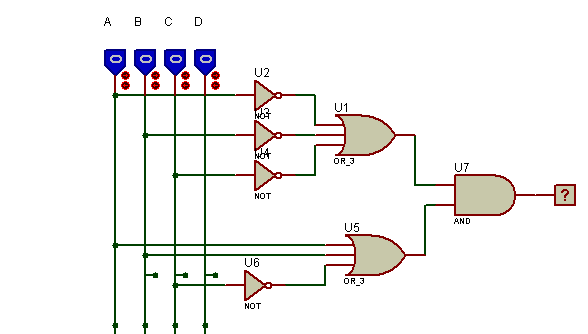
𝑆 = (~𝐴 ∙ 𝐷 + ~𝐵 ∙ ~𝐶 ∙ 𝐷) ∙ (~𝐴 ∙ ~𝐵)

𝑆 = (~𝐴 ∙ ~𝐵 ∙ 𝐷 + ~𝐵 ∙ ~𝐶 ∙ 𝐷 ∙ ~𝐴)

𝑆 = ~𝐴 ∙ ~𝐵 ∙ 𝐷



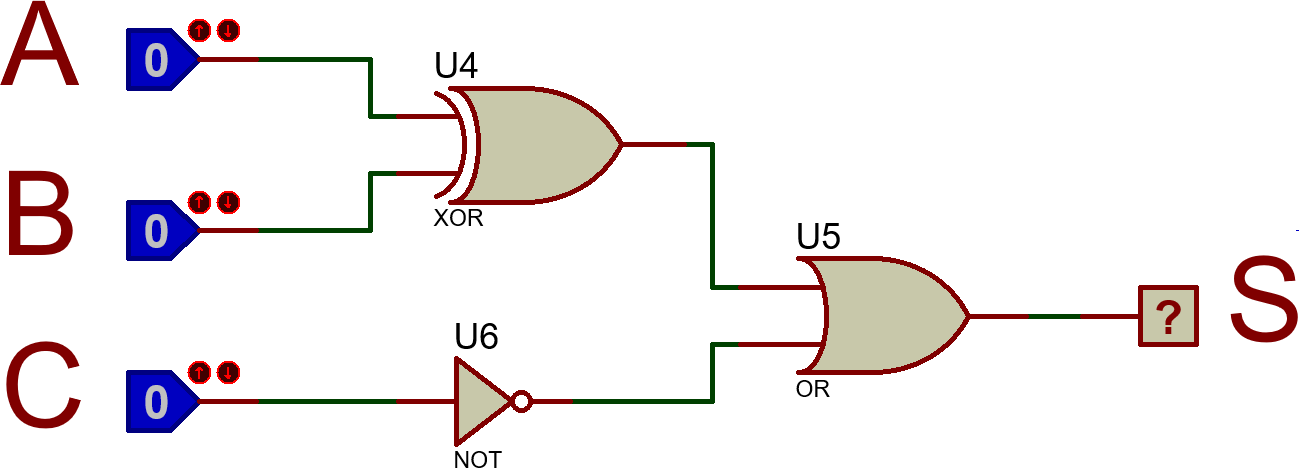
B) 𝑆 = (~𝐴 + ~𝐵 + ~𝐶) ∙ (𝐴 + 𝐵 + ~𝐶)

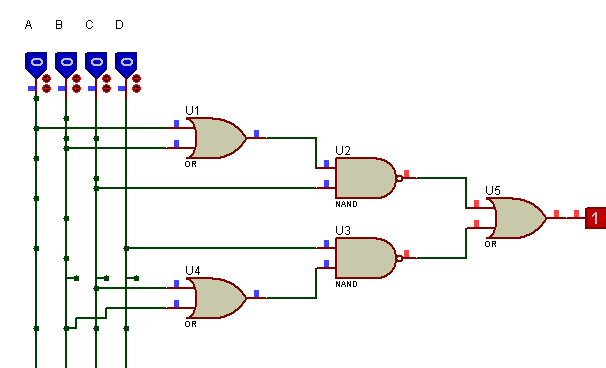


𝑆 = (~𝐴 ∙ 𝐵 + ~𝐴 ∙ ~𝐶 + 𝐴 ∙ ~𝐵 + 𝐴 ∙ ~𝐶 + ~𝐵 ∙ ~𝐶 + 𝐵 ∙ ~𝐶 + ~𝐶)

𝑆 = ~𝐶 + ~𝐴 ∙ 𝐵 + 𝐴 ∙ ~𝐵

𝑆 = ~𝐶 + 𝐴 ⨁ 𝐵

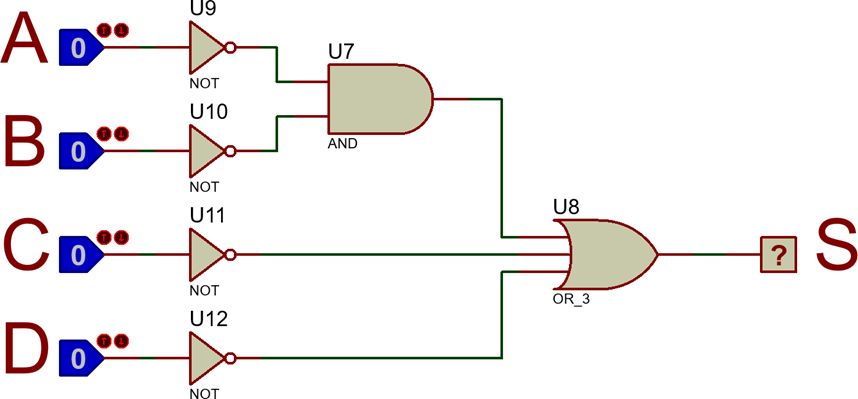
C) 𝑆 = {~[(𝐴 + 𝐵) ∙ 𝐶]} + {~[𝐷 ∙ ( 𝐶 + 𝐵)]}



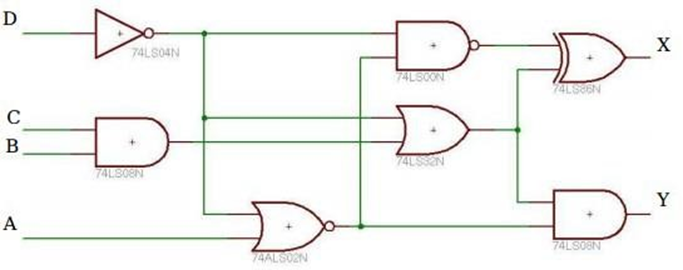
𝑆 = {~(𝐴 + 𝐵) + ~𝐶} + {~𝐷 + ~(𝐶 + 𝐵)}

𝑆 = {(~𝐴 ∙ ~𝐵) + ~𝐶} + {~𝐷 + ~𝐶 ∙ ~𝐵}

𝑆 = ~𝐴 ∙ ~𝐵 + ~𝐶 + ~𝐷



**Questão 2**

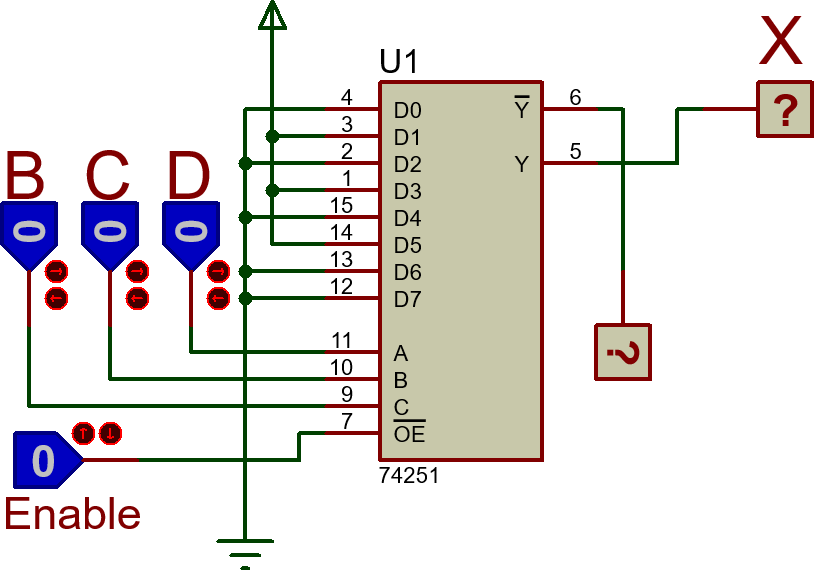


Equações equivalentes ao circuito

𝑋 = (~𝐵 ⋅ ~𝐶) ⋅ 𝐷

𝑌 = ~𝐴 ⋅ 𝐵 ⋅ 𝐶 ⋅ 𝐷

**Equação de X:**

**Multiplexador simulado no Proteus****:**

**Implementação no Verilog (com modulo de simulação):**

**(Modulo de Mux)**

module mux\_8\_1 (Vcc, GRD, B, C, D, X);

input B, C, D, Vcc, GRD;

output X;

reg X;

always @ (\*) begin

case ({B, C, D}) //mux\_8\_1 3'b000: X = GRD;

3'b001: X = Vcc; 3'b010: X = GRD;

3'b011: X = Vcc; 3'b100: X = GRD;

3'b101: X = Vcc; 3'b110: X = GRD;

3'b111: X = GRD;

endcase

end

endmodule

**(Modulo de Simulação)**

module quest2\_mux\_8\_1; reg t\_B;

reg t\_C;

reg t\_D; reg t\_Vcc; reg t\_GRD;

wire t\_Xis;

initial begin: simul\_stop #8 $stop;

end

initial begin

$display("Time \t t\_B \t t\_C \t t\_D \t X");

$monitor(" %0d \t %b \t %b \t %b \t %b",

$time, t\_B, t\_C, t\_D, t\_Xis);

end

initial begin: start t\_B = 0;

t\_C = 0;

t\_D = 0;

t\_Vcc = 1;

t\_GRD = 0;

end

always begin: process

#1 {t\_B, t\_C, t\_D} = {t\_B, t\_C, t\_D} + 1;

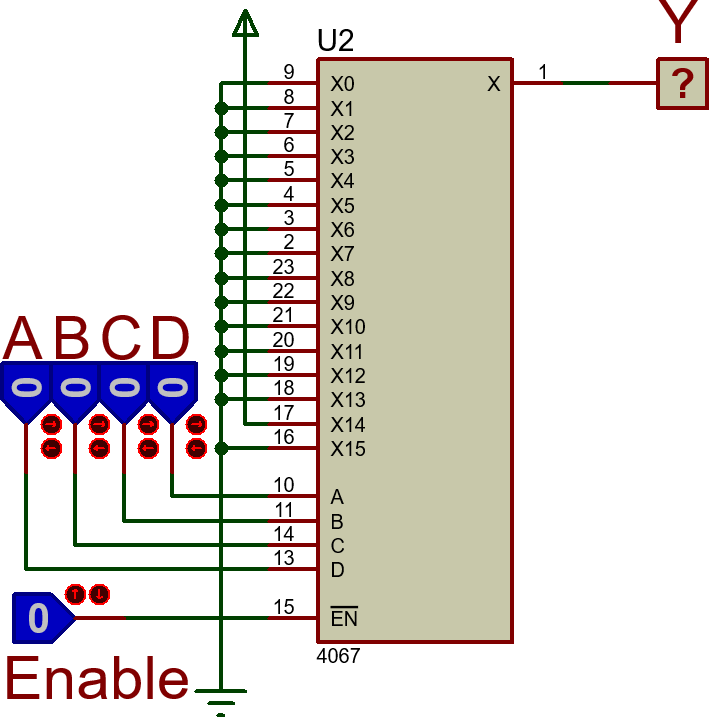
end

mux\_8\_1 simul (.B(t\_B), .C(t\_C), .D(t\_D), .GRD(t\_GRD), .Vcc(t\_Vcc), .X(t\_X));

endmodule

**Equação de Y:**

**Multiplexador simulado no Proteus:**



**Implementação no Verilog (com modulo de simulação):**

**(Modulo do Mux)**

module mux\_16\_1 (Vcc, GRD, A, B, C, D, Y);

input A, B, C, D, Vcc, GRD; output reg Y;

always @ (\*) begin

case ({A, B, C, D}) //mux\_16\_1 4'b0000: Y = GRD;

4'b0001: Y = GRD;

4'b0010: Y = GRD;

4'b0011: Y = GRD;

4'b0100: Y = GRD;

4'b0101: Y = GRD;

4'b0110: Y = GRD;

4'b0111: Y = GRD;

4'b1000: Y = GRD;

4'b1001: Y = GRD;

4'b1010: Y = GRD;

4'b1011: Y = GRD;

4'b1100: Y = GRD;

4'b1101: Y = GRD;

4'b1110: Y = Vcc;

4'b1111: Y = GRD;

Endcase

end

endmodule

**(Modulo de Simulação)**

module quest2\_mux\_16\_1;

reg t\_A; reg t\_B; reg t\_C; reg t\_D; reg t\_Vcc; reg t\_GRD;

wire t\_Y;

initial begin: simul\_stop

#16 $stop;

end

initial begin

$display("Time \t t\_A \t t\_B \t t\_C \t t\_D \t Y");

$monitor(" %0d \t %b \t %b \t %b \t %b \t %b",

$time, t\_A, t\_B, t\_C, t\_D, t\_Y);

end

initial begin: start

t\_A = 0;

t\_B = 0;

t\_C = 0;

t\_D = 0;

t\_Vcc = 1;

t\_GRD = 0;

end

always begin: process

#1 {t\_A, t\_B, t\_C, t\_D} = {t\_A, t\_B, t\_C, t\_D} + 1;

end

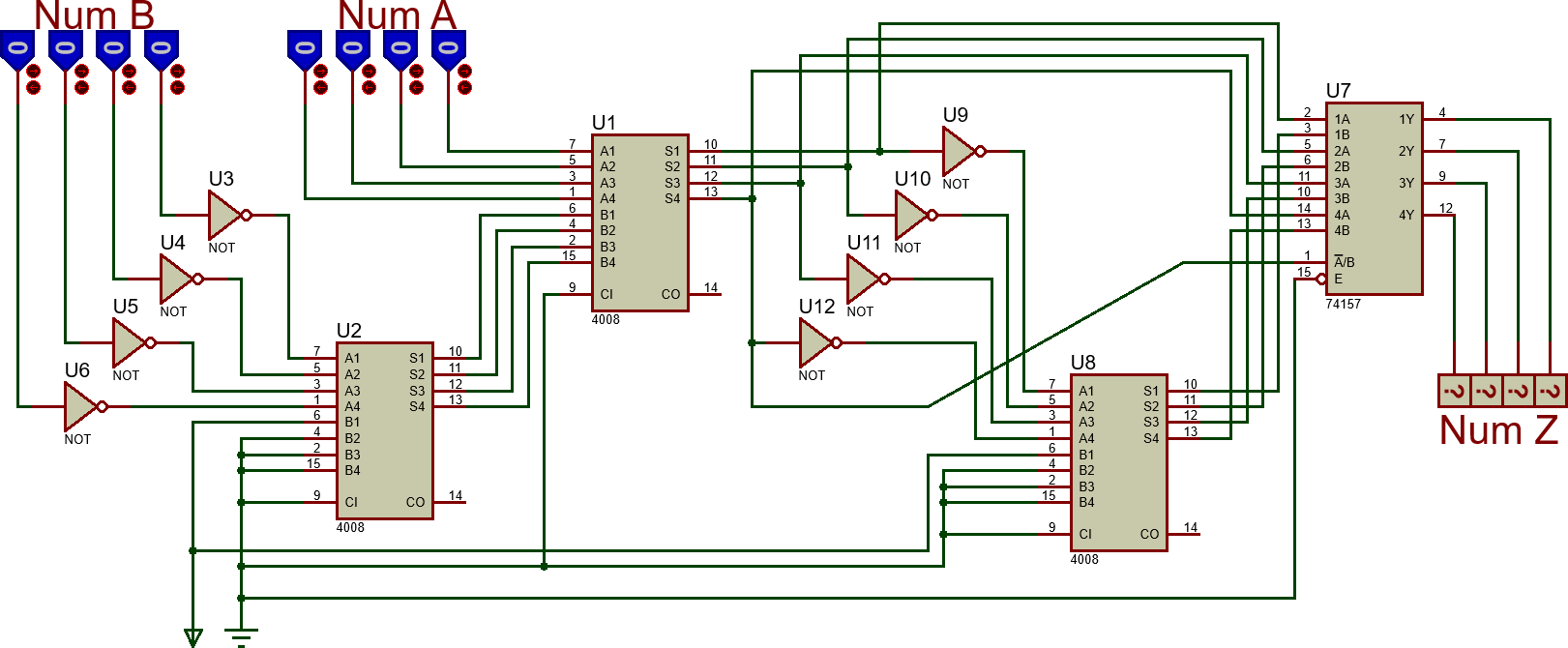
mux\_16\_1 simul (.A(t\_A), .B(t\_B), .C(t\_C), .D(t\_D), .GRD(t\_GRD), .Vcc(t\_Vcc), .Y(t\_Y));

endmodule

**Questão 5**

Implementar a função Z = |A – B|

**Implementação no Proteus:**



**Implementação no Verilog:**

module somaSub(A, B, Ze);

input [3:0] A, B;

output reg [3:0] Ze;

always @ ( \* ) begin

if (A >= B)

Ze = A - B;

else

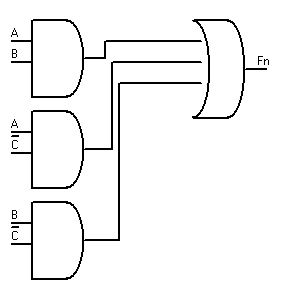
Ze = B - A;

end

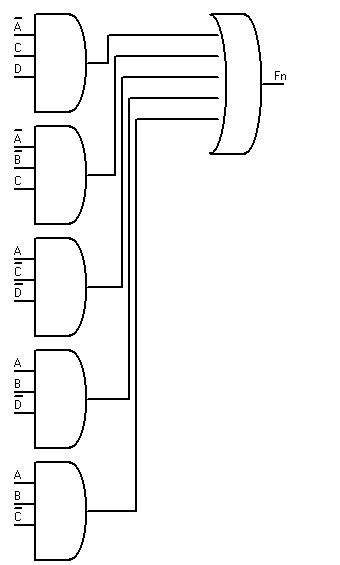
endmodule

**Questão 9**

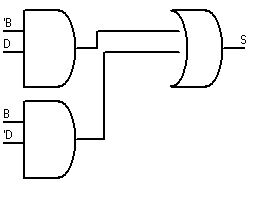
A.B + A.~C + B.~C



~A.C.D + ~A.~B.C + A.~C.~D + A.B.~C



~B.D + B.~D



**Onde A e B são as posições do elevador e C e D as posições do usuário.**

A implementação em Verilog:

module elevador(E, U, S);

input [1:0] E, U;

output reg [2:0] d;

always @ ( \* ) begin

S[0] = (A[0] & A[1]) | (A[0] & ~B[0]) + (A[1] & ~B[0]);

S[1] = (~A[0] & B[0] & B[1]) | (~A[0] & ~A[1] & B[0]) | (A[0] & ~B[0] & ~B[1]) | (A[0] & A[1] & ~B[0]);

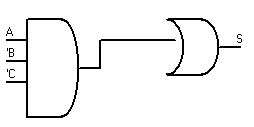
S[2] = (~A[1] & B[1]) | (A[1] & ~B[1]);

end

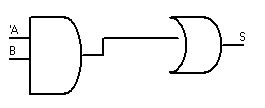
endmodule

**Questão 11**

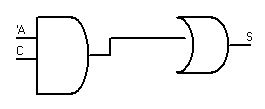
A.~B.~C



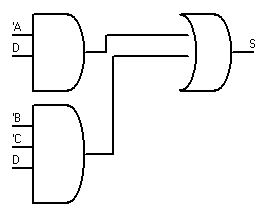
~A.B

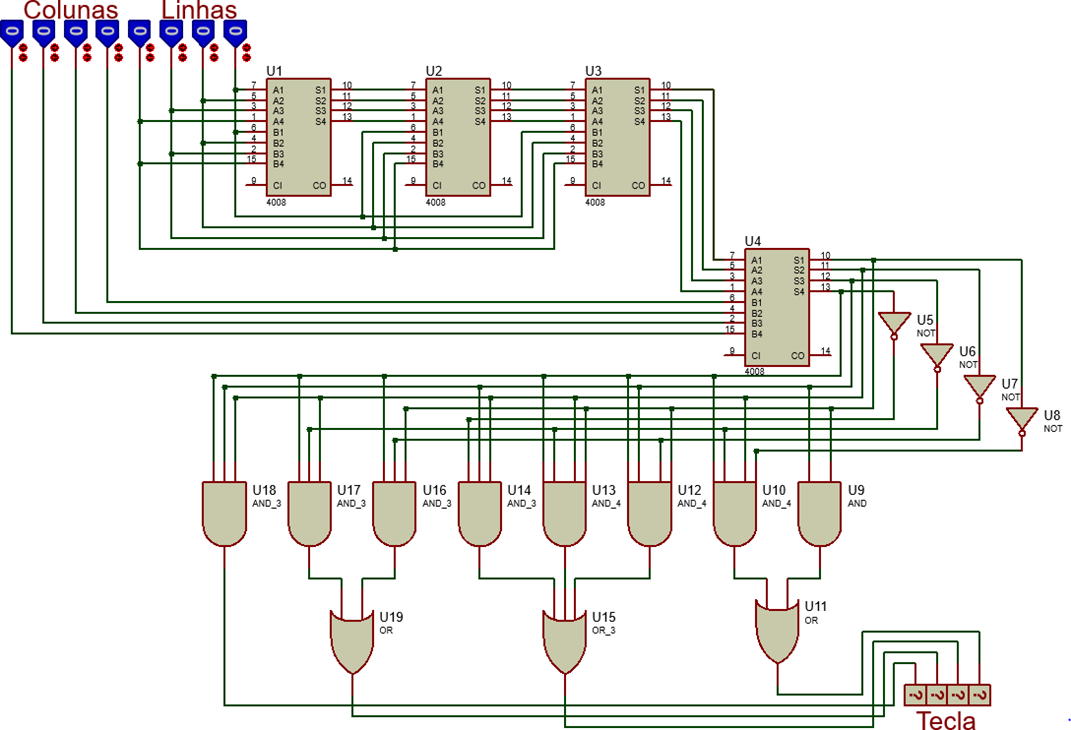


~A.C



~A.D + ~B.~C.D





**Questão 12**

W = 'A\*B + C\*'A + D\*'A + 'C\*'D\*A\*'B

X = 'D\*'A\*B + C\*D\*'A + 'C\*'D\*'A + 'C\*D\*A\*'B

Y = D\*'A\*B + 'C\*'A\*B + 'C\*'D\*'B

Z = 'C\*A\*'B + C\*'A\*B + 'D\*'A\*'B

